AMENDMENTS TO THE CLAIMS

Applicants submit below a complete listing of the current claims, including marked-up claims with insertions indicated by underlining and deletions indicated by strikeouts and/or double bracketing. This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of the Claims

1. (Currently amended) A ~~library of software program products, the library comprising a set of~~ computer readable storage medium encoded with software routines for use by an embedded software application requiring software (SW) protocol layers, profiles and/or application code embedded on a processor, the ~~library further comprising software for~~ software routines, when executed, providing an interface between the embedded software application running on the processor and the SW protocol layers and/or the profiles and/or the application code, wherein the interface and the SW protocol layers ~~communicating~~ communicate towards an underlying operating system (OS) through an abstraction layer that maps OS-independent function calls to OS-specific function calls, and wherein the interface assigns priorities to tasks of the embedded software application such that the tasks of the embedded software application are interrupted by OS tasks.

2. (Currently amended) The ~~library~~ computer readable storage medium according to claim 1, wherein the interface is between the embedded software application running on the processor and a telecommunications module.

3. (Currently amended) The ~~library~~ computer readable storage medium according to claim 2, wherein the telecommunications module is a lower layer SW protocol of a ~~Bluetooth~~ short range wireless device.

4. (Currently amended) The ~~library~~ computer readable storage medium according to claim 2, wherein the interface uses telecommunications controller interface communications.

5. (Currently amended) The ~~library~~ computer readable storage medium according to claim 4, wherein the communications are Host Controller Interface (HCI) communications for communication with the telecommunications module.

6. (Currently amended) The ~~library~~ computer readable storage medium according to claim 2, wherein the interface allows the embedded software application to communicate[[s]] with [[a]] the telecommunications module for executing a telecommunications protocol.

7. (Currently amended) The ~~library~~ computer readable storage medium according to claim 6, wherein the interface allows the embedded software application to communicate[[s]] with a hardware input/output interface.

8. (Canceled)

9. (Currently amended) The ~~library~~ computer readable storage medium according to claim [[8]] 1, wherein the computer readable storage medium is a CD-ROM or DVD-ROM or a memory or data storage device.

10. (Currently amended) A telecommunications device comprising an interface executing on the telecommunications device towards an underlying operating system (OS), layers of a telecommunications protocol and any hardware available for an embedded application, the interface and the layers of the telecommunications protocol communicating towards the underlying OS through an abstraction layer that maps OS-independent function calls to OS-specific function calls, wherein the interface assigns priorities to tasks of at least one software application executing on the telecommunications device, such that the tasks of the at least one software application are interrupted by OS tasks.

11. (Original) The telecommunications device according to claim 10 wherein the interface communicates with the telecommunications protocol via telecommunications controller interface communications.

12. (Previously presented) The telecommunications device according to claim 10, wherein the interface is an Application Programming Interface (API).

13. (Currently amended) Host processing system for executing the ~~library of computer programs~~ software routines encoded on the computer readable storage medium in accordance with claim 1.

14. (Currently amended) An Application Programming Interface (API) comprising a plurality of commands that, when executed by a processor, provide[[s]] functions to a software application requiring software (SW) protocol layers, profiles and/or application code embedded on [[a]] the processor, the API communicating towards an underlying operating system (OS), layers of a telecommunications protocol and any hardware available for an embedded application, the API and the layers of the telecommunications protocol communicating towards the underlying OS through an abstraction layer that maps OS-independent function calls to OS-specific function calls, wherein the API assigns priorities to tasks of the software application such that the tasks of the software application are interrupted by OS tasks.

15. (Previously presented) The API according to claim 14, wherein the API communicates with the protocol layers using Host Controller Interface (HCI) communications.

16. (Previously presented) The API of claim 14, stored on a computer readable medium.

17. (Currently amended) A method of embedding a software application requiring software (SW) protocol layers, profiles and/or application code embedded on a processor, the method comprising generating an Application Programming Interface (API) for communicating towards at least one of an underlying operating system (OS), layers of a telecommunications

protocol and any hardware available for an embedded application, wherein the API ~~and the layers of the telecommunications protocol communicating~~ communicates towards the underlying OS through an abstraction layer [[that]] through which the layers of the telecommunications protocol also communicate towards the underlying OS, wherein the abstraction layer maps OS-independent function calls to OS-specific function calls, and wherein the API assigns priorities to tasks of the software application such that the tasks of the software application are interrupted by OS tasks.

18. (Currently amended) A method of operating a telecommunications device, the method comprising executing, via at least one processor of the telecommunications device, an interface towards an underlying operating system (OS), layers of a telecommunications protocol and any hardware available for an embedded application, the interface and the layers of the telecommunications protocol communicating towards the underlying OS through an abstraction layer that maps OS-independent function calls to OS-specific function calls, wherein the interface assigns priorities to tasks of at least one software application executing on the telecommunications device, such that the tasks of the at least one software application are interrupted by OS tasks.

19. (Previously presented) The method according to claim 18, wherein the interface is an Application Programming Interface (API).

20. (Currently amended) A telecommunications device comprising layers of a telecommunications protocol and an interface executing on the telecommunications device towards an underlying operating system (OS), the interface and the layers of the telecommunications protocol communicating towards the underlying OS through an abstraction layer that maps OS-independent function calls to OS-specific function calls, wherein the interface assigns priorities to tasks of at least one software application executing on the telecommunications device, such that the tasks of the at least one software application are interrupted by OS tasks.

21. (Previously presented) The telecommunications device according to claim 20, wherein the interface is an Application Programming Interface (API).

22. (Currently amended) A method of embedding into a telecommunications device a software application requiring software (SW) protocol layers, profiles and/or application code embedded on a processor of the telecommunications device, the method comprising generating an Application Programming Interface (API) for communicating towards an underlying operating system (OS), wherein the API ~~and the SW protocol layers communicating~~ communicates towards the underlying OS through an abstraction layer [[that]] through which the SW protocol layers also communicate towards the underlying OS, wherein the abstraction layer maps OS-independent function calls to OS-specific function calls, and wherein the API assigns priorities to tasks of the software application such that the tasks of the software application are interrupted by OS tasks.

23. (Currently amended) A method of operating a telecommunications device comprising executing, via at least one processor of the telecommunications device, layers of a telecommunications protocol and an interface towards an underlying operating system (OS), the interface and the layers of the telecommunications protocol communicating towards the underlying OS through an abstraction layer that maps OS-independent function calls to OS-specific function calls, wherein the interface assigns priorities to tasks of at least one software application executing on the telecommunications device, such that the tasks of the at least one software application are interrupted by OS tasks.

24. (Previously presented) The method according to claim 23, wherein the interface is an Application Programming Interface (API).